Information Sciences 430–431 (2018) 444–466

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

How to make a neural network say "Don't know"

Bikram Karmakar[a], Nikhil R. Pal[b,*]

[a] Department of Statistics, University of Pennsylvania, Philadelphia, PA 19104-6340, USA
[b] Electronics and Communication Sciences Unit, Indian Statistical Institute, Calcutta 700 108, India

ARTICLE INFO

Article history:
Received 18 April 2017
Revised 27 November 2017
Accepted 30 November 2017

Keywords:
Classification algorithms
Learning
Multi-layer neural network
Multi-layer perceptrons
Support estimation
Regularization

ABSTRACT

Despite various advantages, due to improper training, sometimes a Multi-Layer Perceptron (MLP) classifies a test data point far from the training data into a completely irrelevant class. On the other hand, for example, when we train an MLP to distinguish between four types of childhood cancers (neuroblastoma, rhabdomyosarcoma, non-Hodgkin lymphomas, and Ewing sarcoma) using gene expression profiles and test it on some other kind of cancer (or normal patient) data, the test data will be misclassified to one of the four classes. These unexpected situations arise due to the "open world" nature of the problem. Such problems exist with many other learning systems. We want to address these problems by equipping the network with the ability to not make any judgment when it should not. We have developed an algorithm to provide a practical solution to this problem. We first estimate the domain of the training data (sampling window). We show consistency of our estimate along with some other interesting results. An MLP should say "Don't know" if a test point is outside the sampling window. To realize this, we generate observations from the complement region and label them to a new class "Don't know". We train a network with training data along with the generated data to get a classifier that supports the extra class. The problem of generating a large number of points from the complement region is dealt with a novel scheme that exploits the input sensitivity of the system on the complement region via a regularization. We study the effectiveness of our proposed method with several data sets.

1. Introduction

The task of a classifier is to assign a class label to a feature vector with an unknown class. Typically the classifier learns the decision rule using a training data set. The Multilayered Perceptron (MLP) is one such family of classifiers, which is extensively used to solve numerous problems in a variety of domains, e.g., in atmospheric sciences [19], medicine [33], speech recognition [30] and statistical sciences [8]. MLPs are also extensively used as function approximators or predictors. It has been proved by Hornik et al. [26] that MLPs with a single hidden layer can act as a universal approximator for a sufficiently large class of functions. Also, the learning algorithms for MLPs are fast, which promote it as an industrially usable tool for data mining. But MLPs have the disadvantage of being not interpretable. It is difficult to understand the resultant classifier in physical terms. This lack of readability often limits the use of MLPs on critical applications, such as diagnosis of cancer. Standard methods of training MLPs involve phases of training, validation and testing depending on the data set at hand (see [3]).

* Corresponding author.
E-mail addresses: bikramk@wharton.upenn.edu (B. Karmakar), nikhil@isical.ac.in (N.R. Pal).

https://doi.org/10.1016/j.ins.2017.11.061
0020-0255/

Almost every classification problem is an open-world problem, where we have knowledge about a few classes as expressed by a finite training data set. This results in several problems. First, at the time of testing, there could be data from some unknown classes and a classifier, such as a neural network, will assign such a data point to one of the existing classes. Moreover, the training data may not be adequate to represent the complete picture of all the classes under consideration and consequently the classifier may generalize poorly. The classifier may also over-fit (memorize) the data and in that case, it will not generalize well. The statistical characteristics of data defining a class (concept) may change with time (concept drift) [18,44]. In this case also the classifier may not generalize well. The problems of poor generalization and over-fitting stem primarily from the fact that neural networks are trained with a finite training sample. The training data are reused in every epoch and as a result, the neural network concentrates more and more on these points and often results in a bad generalization. When the network has more degrees of freedom than what is needed, it may memorize the training data which may result in poor generalization.

So there are at least four problems:

1. In an open-world situation, a test data may come from an unknown class and will get misclassified.
2. In a closed-world situation, a test data point may come from outside the "sampling window". In this case, the network should not make any decision, but it will assign some class.
3. In a closed-world situation, a test data point may come from outside the "sampling window". The class assigned by the network may be unrealistic in the sense that the test data may be located close to the training data of class i, but the assigned class is j, and $i \neq j$.
4. In a closed-world situation, the statistical properties of one or more class may change with time (concept drift). If such a drift is significant, the network performance will fall and the network should not make any decision for drifted data.

The above four problems are different in nature, but all have one thing in common: these problems arise when the test data point is outside the sampling window of the training data. So, a common solution to all four is to make the network learn to say "Don't Know" when the test data point is outside the sampling window. That is what we shall try to do.

Many researchers have tried to address some of these problems [6,7,10,11,14,22,27,36,37]. In our view, these problems can be addressed, at least, in three different ways: First, based on some preprocessing (this may involve another machine learning system), one may discard a test data point as not coming from the classes represented in the training data [36,37]. Second, after the classifier is trained, one can analyse its output and make a decision whether to reject the test data or not [10,11,14,22]. Third, explicitly one may try to model the sampling window of the training data and then generate samples from the complement world and train a classifier with an additional class called "Reject" or "Don't know" [6,7]. In the present study we shall follow the last approach. Next we provide a summary of several methods for rejecting a data point.

The work of Chow [10] has been a classic work in the field. The problem has been stated as a standard classification problem with an option to "reject" a test data point, that is not to assign a test data point to any of the known classes under appropriate conditions. Chow's approach models the problem as a decision problem. This method needs to assume a model for each class, $p(\mathbf{x} \mid i)$, the probability of $\mathbf{x}$ coming from class i. Moreover, we also need to assume the priors p_i for each class. Then for a given loss function, a non-randomized decision rule is used.

The convention is to use class label 0 for "rejecting" a data point. Suppose the loss of classifying a point in class i as an element in class j is w_{ij}. Let the classes be numbered $1, 2, \ldots, c$. Then the best decision rule is given by Chow [10]

$$\text{label}\,\mathbf{x}\,\text{as}\,k = \arg\min_{0 \le j \le c} \left[\sum_{i=1}^{c} (w_{ij} - w_{i0}) \cdot p_i \cdot p(\mathbf{x} \mid i) \right] I(j \neq 0).$$

In a later work, Chow uses a different argument [11]. The aim there is to get a decision rule which rejects a pattern if the maximum of the posterior probabilities is less than a threshold. In this case also there is a best rule for classification, given as follows: For a data point $\mathbf{x}$

1. "Reject" if

$$\max_i \; p_i \cdot p(\mathbf{x} \mid i) < (1 - t) \sum_{i=1}^{c} p_i \cdot p(\mathbf{x} \mid i).$$

2. If not rejected then label it in class k if

$$k = \arg\max_{1 \le i \le c} \{p_i \cdot p(x \mid i)\}.$$

Here $t > 0$ is a threshold. The optimality of this rule is in terms of achieving the minimum misclassification probability in known classes. This rule defines the error-reject trade-off. If w_e, w_r, w_c are the costs associated with erroneous, reject, and correct decisions, respectively, then t is suggested as $t = \frac{w_r - w_c}{w_e - w_c}$.

In a follow-up paper Dubuisson and Masson [14] studied the same framework. They pointed out a gap in the previous model. If a test data point comes from an area which is far from any training data then that data point must not be classified, even if the maximum of the posterior probability is greater than the threshold. Dubuisson and Masson provided a simple fix to this problem. They would also "reject" a point $\mathbf{x}$, if $\sum_{i=1}^{c} p_i \cdot p(\mathbf{x} \mid i)$ is below some prefixed threshold. There